BIG

Revolutionizing how customers build, deploy and operate highly available and scalable services

Application Platform Infrastructure Team
Windows Server Product Group
Microsoft Corporation

June 18, 2002

Agenda

- Vision
- **Architecture**
  - *Hardware Reference Platform*
  - *Resource Management*
  - *IBIG*
  - *Service Definition Model (SDM)*
  - *Operations Logic*
- **Customers and Scenarios**
- **Wrap-up**

The Broad Vision

- **Enable**
  - ***development*** *of distributed, scalable and highly available services using Visual Studio and* ***reusable*** *building blocks like SQL, IIS.*
  - ***deployment*** *across a set of* ***abstracted*** *hardware resources that are automatically allocated, purposed and configured*
  - ***lower cost of ownership*** *through automation by codifying operational best practices to control service availability and growth*
  - ***procurement*** *of standardized data center hardware that leverages* ***commodity economics***

BIG is a revolutionary re-think of the way highly available and scalable services are built, deployed, operated

BIG Will Deliver a Services Platform

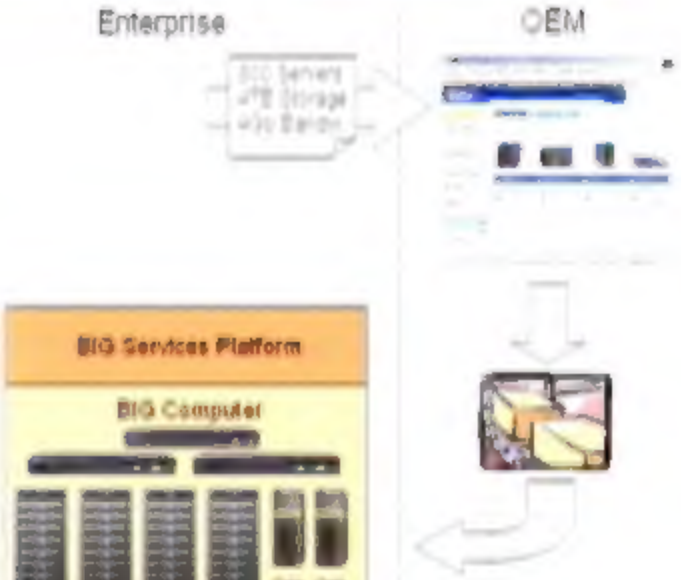
- **Hardware reference platform that aggregates commodity hardware to build a single large computer that we call the BIG Computer**
  - *Includes interconnected servers, network devices, and storage*
- **Hardware abstraction layer that virtualizes resources**
  - *Enables dynamic hardware binding and re-deployment and automated network configuration*
- **Service Definition Model (SDM) for developers to describe an entire service**
  - *Enables developers to rapidly build new services using highly available SQL, IIS and other reusable building block components*
  - *Requires incremental changes to BIG enable existing services*
- **Highly available runtime that supports the SDM**
  - *Enables hosting multiple scalable services inside the BIG Computer*
- **Operations logic framework for automating operational best practices**
  - *Enables policy expression and enforcement*

BIG Ecosystem

Enterprise

BIG Ecosystem

BIG Ecosystem



BIG Ecosystem

Developer

Enterprise

OEM

BIG Ecosystem

Developer

ASP.NET

Enterprise

OEM

BIG Ecosystem

Developer

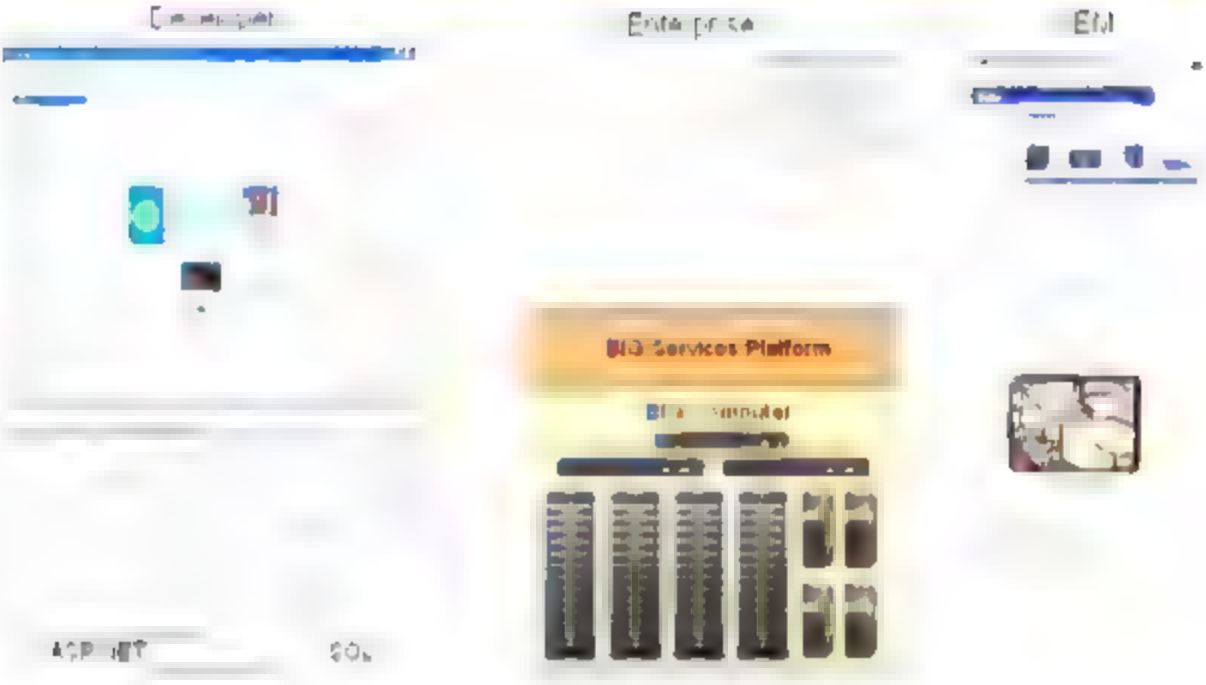
ASP.NET

SQL

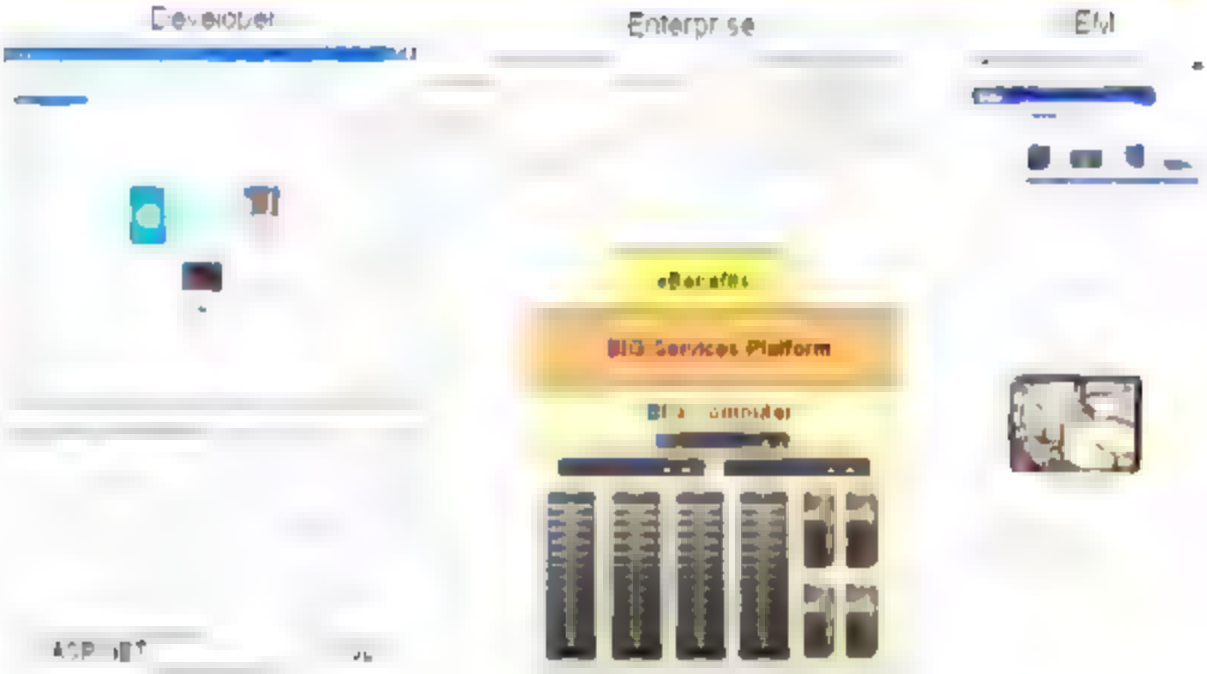
Enterprise

OEM

BIG Ecosystem



BIG Ecosystem



BIG Ecosystem

Developer

Enterprise

OEM

perations Logi

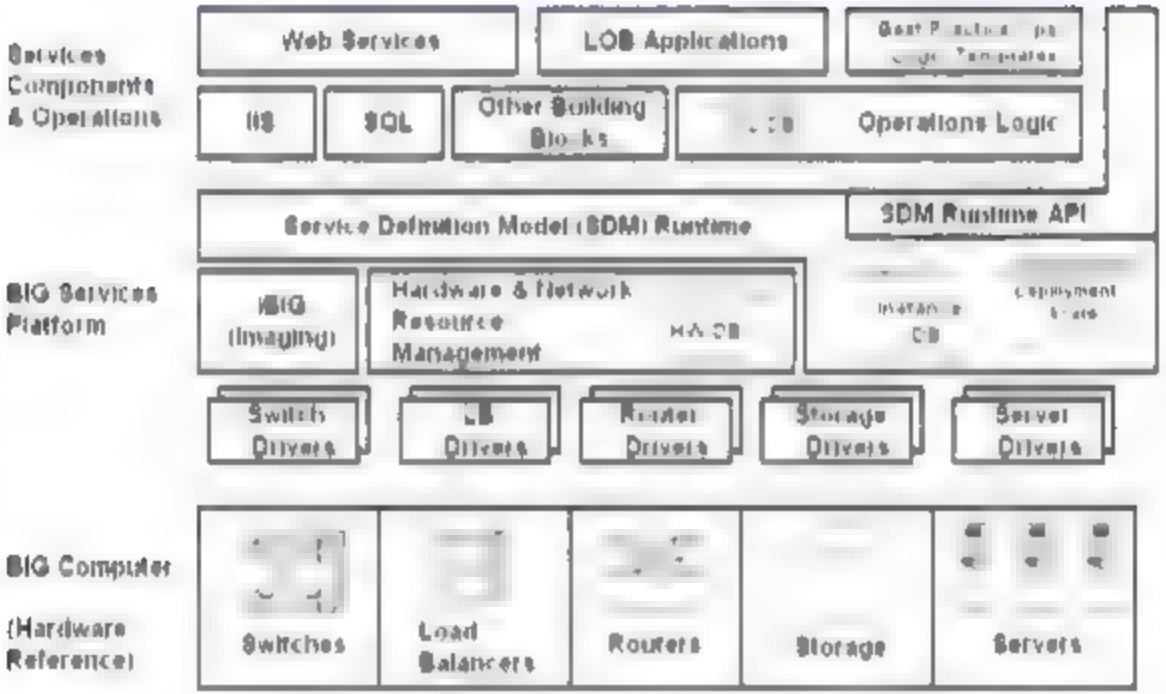
BIG Ecosystem

Developer

Enterprise

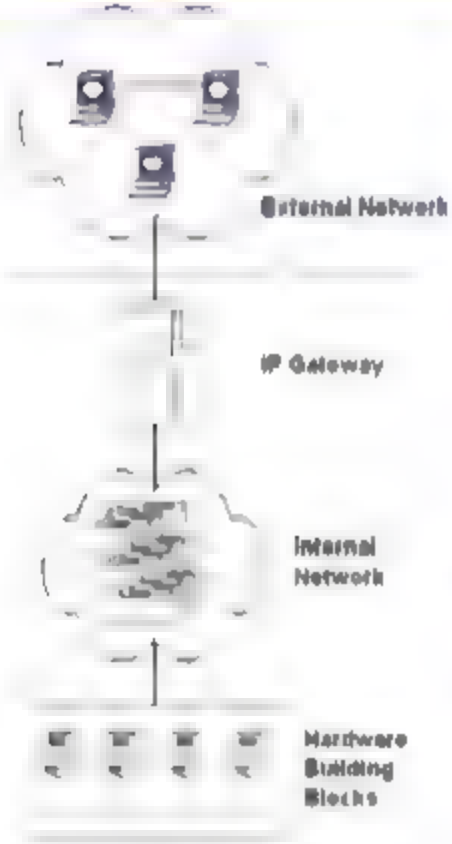
CEM

BIG Services Platform Architecture



Hardware Reference Platform Initiative

Hardware Reference Platform (BIG Computer)



Motivation

- Reduces the cost of design, test and operations
- Limits number of hardware devices to support
- Constrains the network topology and enables automation of network configuration
- Customers adverse to BIG taking over the entire data center (PXE, DHCP, DNS, network switches)

IP Gateway

- [illegible]

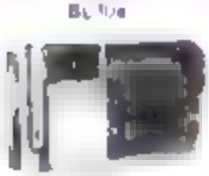
Internal Network

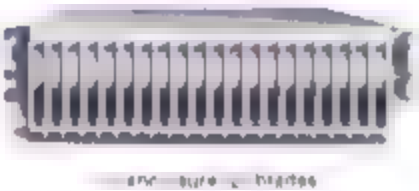
- [illegible]

Hardware Building Blocks

- [illegible]

Examples of Current Products that can be Inside a BIG Computer





Compaq ProLiant BL e-class

IBM BladeCenter

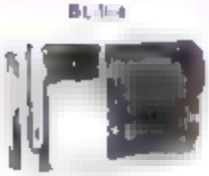
Dell PowerEdge 1655MC

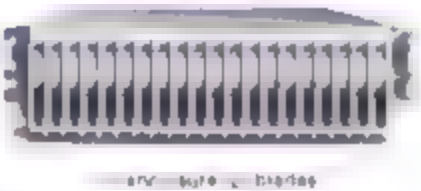
HP bc 1100

Fabric blades

Infrastructure Services

Examples of Current Products that can be Inside a BIG Computer





Compaq ProLiant BL e-class

IBM BladeCenter

Dell PowerEdge 1655MC

HP bc1100

BIG Resource Management

- Global Resource Manager automatically allocates resources and binds services to the BIG Computer
  -
- Resource providers
  -
  -
  -
  -
  -
  -
- Resource Manager optimizes component placement to conserve valuable cross-sectional bandwidth and meet resource constraints
- Resource Manager frees operators from the detailed tasks of resource management
  -
  -

BIG Hardware Resource Discovery and Management

- Dynamic or static hardware discovery
- Resource drivers are bound to hardware devices and expose logical resources for allocation
- Allocation translates logical resource request to physical resource availability

Properties

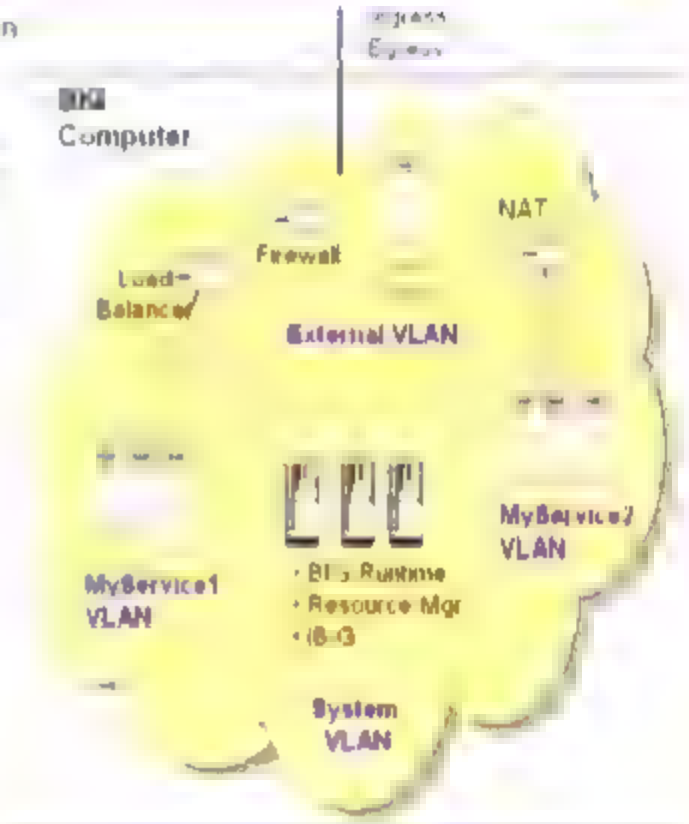
- Power
- Network
- Storage
- Processor
- Memory
- Location

Physical Resource Graph

Physical Resource Database

Network Management within the BIG Computer

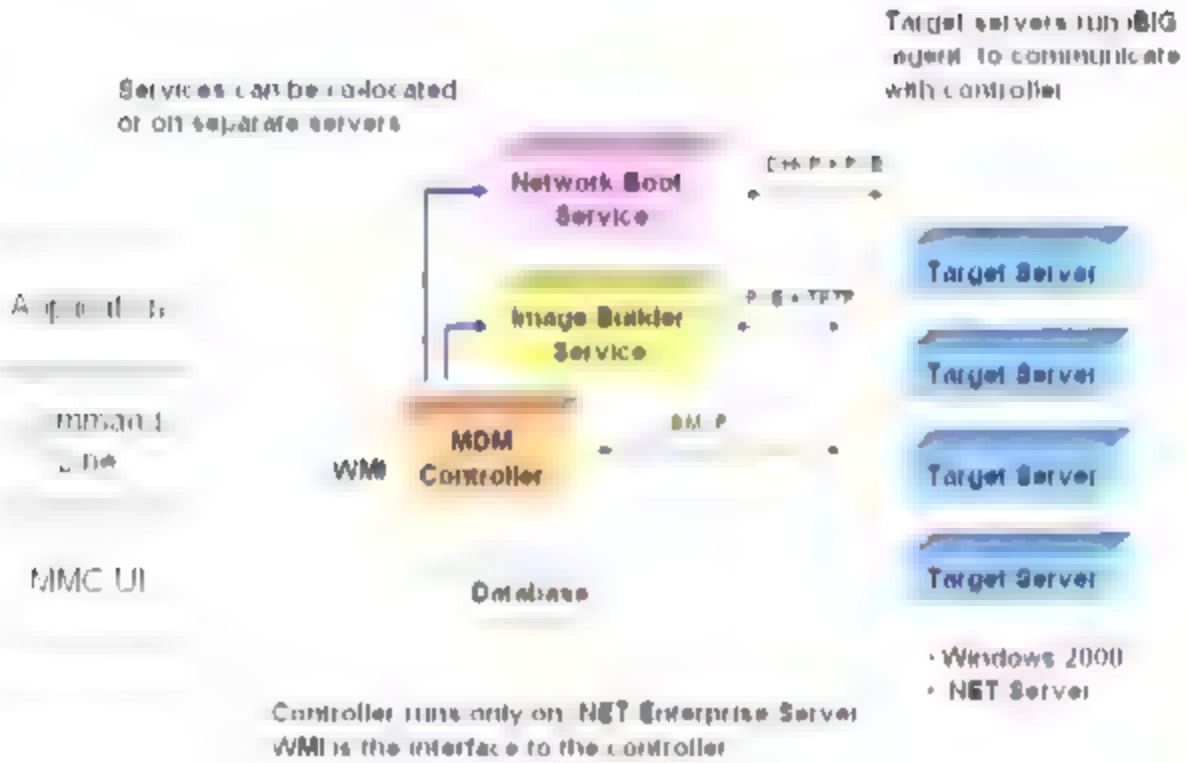
- BIG Computer follows an abstraction layer for network resources
- BIG programs the network switches and load balancers and configures the single software
- vLANs provide isolation
  - [illegible]
- Network resources
  - [illegible]



iBIG Features

- Base Deployment Services
- Image Deployment and Management
- Multiple Device Management (MDM)
- Supports Windows 2000 and .NET Server targets

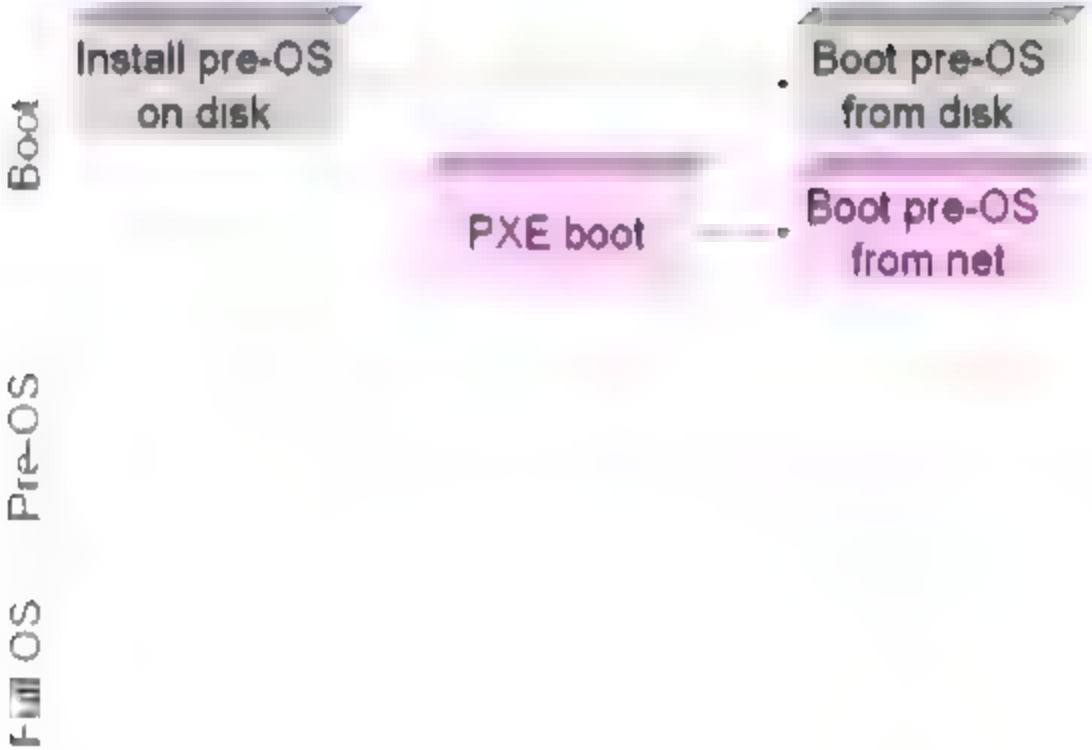
iBIG Architecture



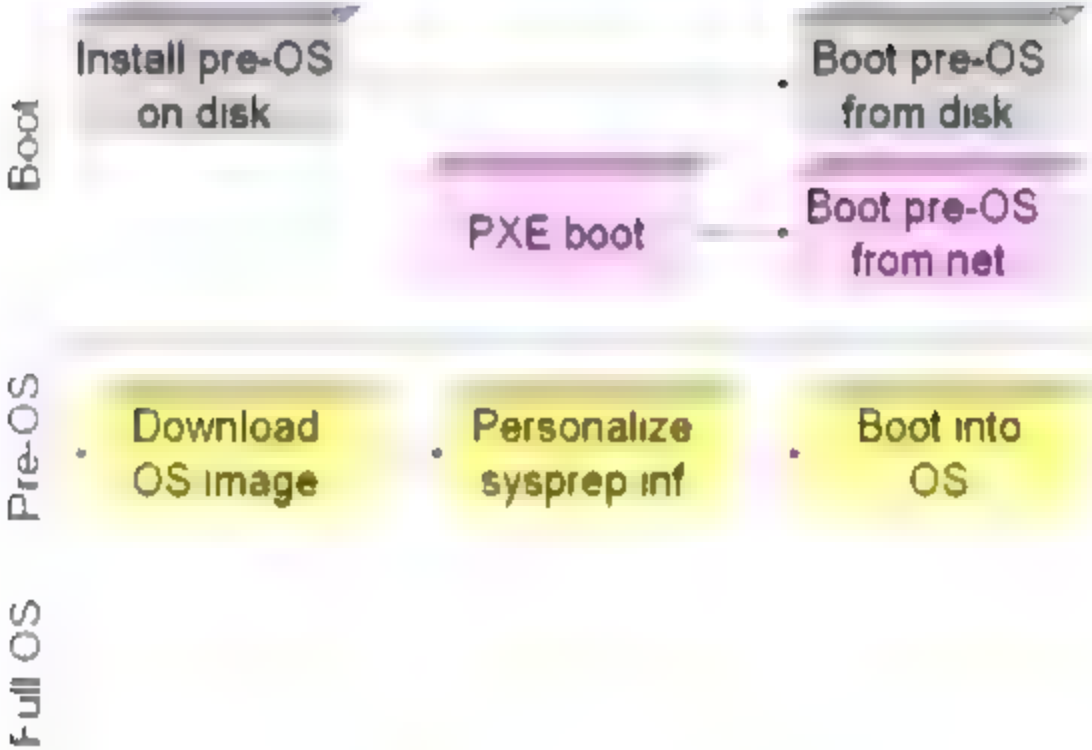
Full OS Pre-OS Boot

Server Purposing

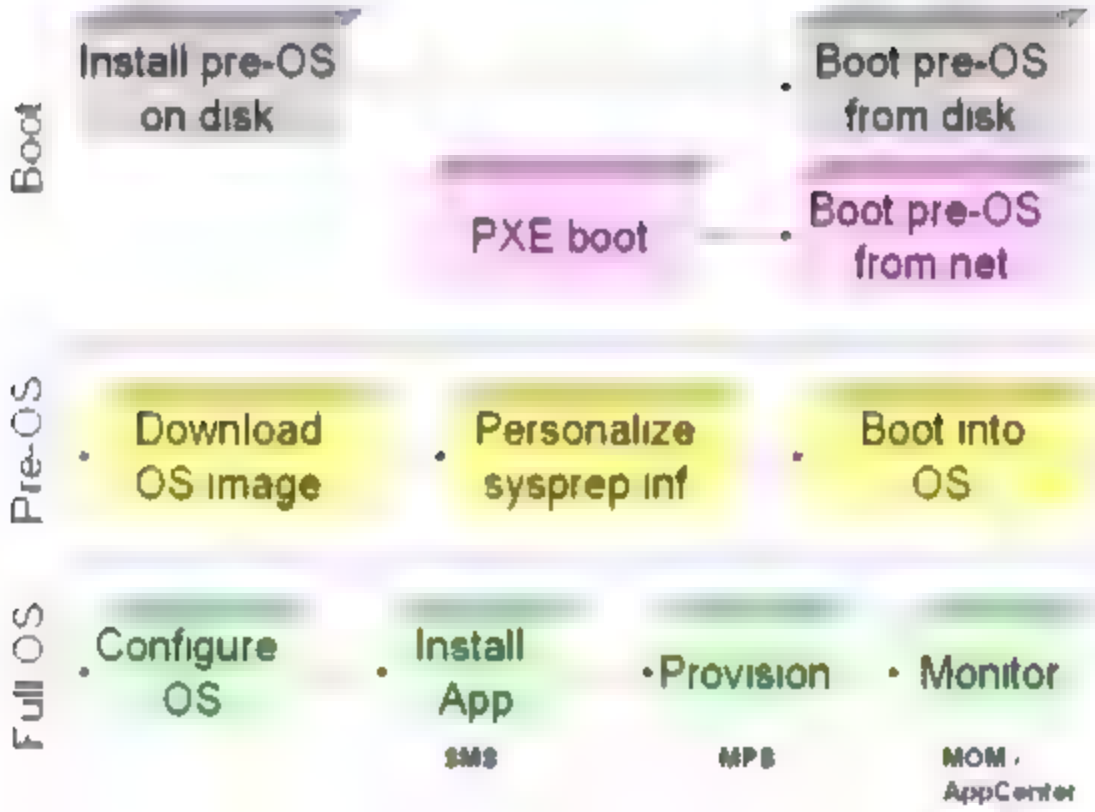
Server Purposing



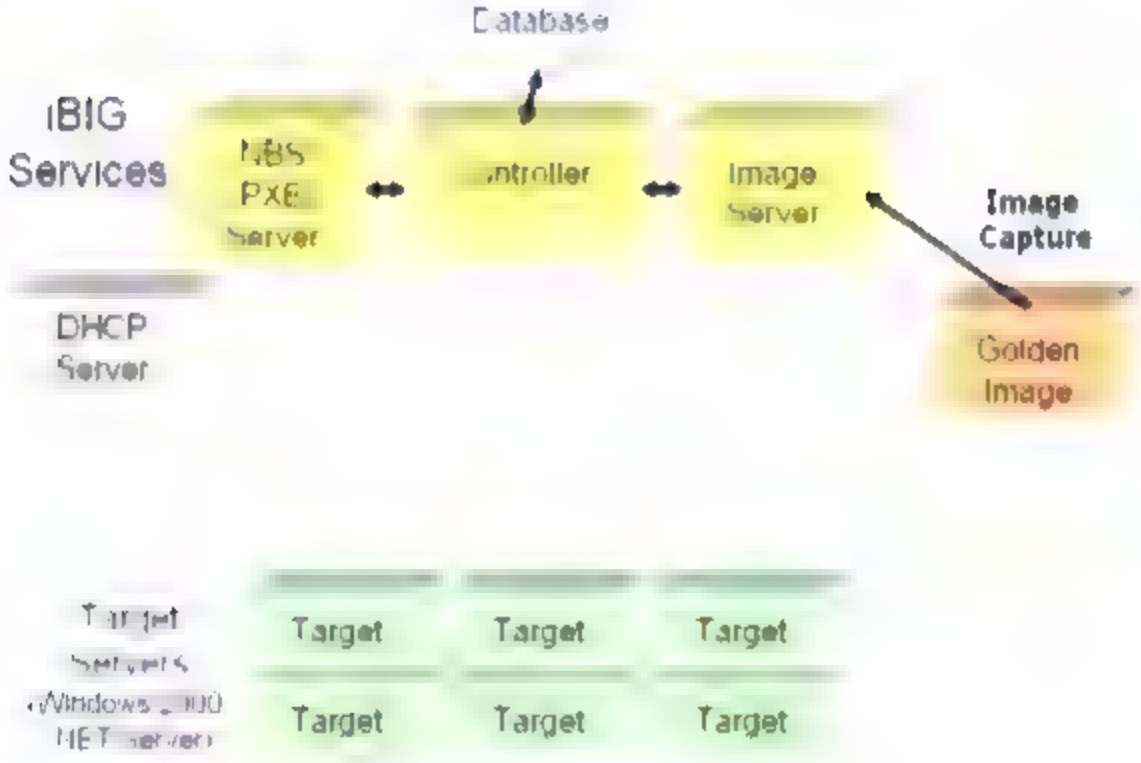
Server Purposing



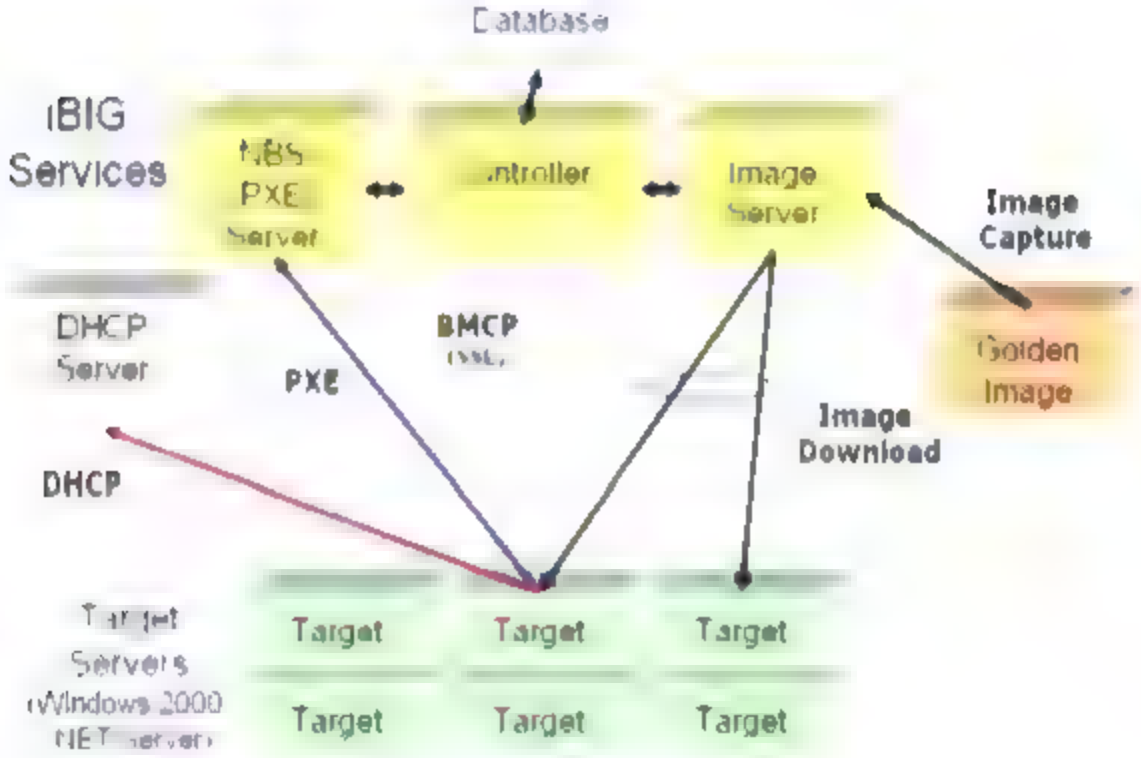
Server Purposing



Remote Boot and Imaging



Remote Boot and Imaging



Service Definition Model (SDM)

- The programmatic description of the entire service
  - Declarative
  - service structure
  - a framework
  - Component-based
- SDML is the declarative language for defining Service Definition Models

Components, Ports and Wires

- Components are units of implementation, deployment and operations
  - [illegible]
  - [illegible]
  - [illegible]
- Ports are names (service access points) with an associated type (protocol)
  - [illegible]
  - [illegible]
- Wires are the permissible bindings between ports
  - [illegible]

Components, Ports and Wires

- Components are units of implementation, deployment and operations
  - [illegible]
  - [illegible]
  - [illegible]
- Ports are names (service access points) with an associated type (protocol)
  - [illegible]
  - [illegible]
- Wires are the permissible bindings between ports
  - [illegible]

SDML Example: MyService.sdml

```
using System
using System.SQL.
using System.IIS.
assembly name MyService
```

```
componenttype MyBackEnd
  SQLDatabase {
  implementation  MySQL MyCLRApp
}
```

```
componenttype MyService
{
  component MyBackEnd be
  port http = fe.http.
  wire TDS fde {
        fe.catalog
        be.sql
  }
  implementation  MyService MyCLRApp
}
```

MyService

MyBackEnd be

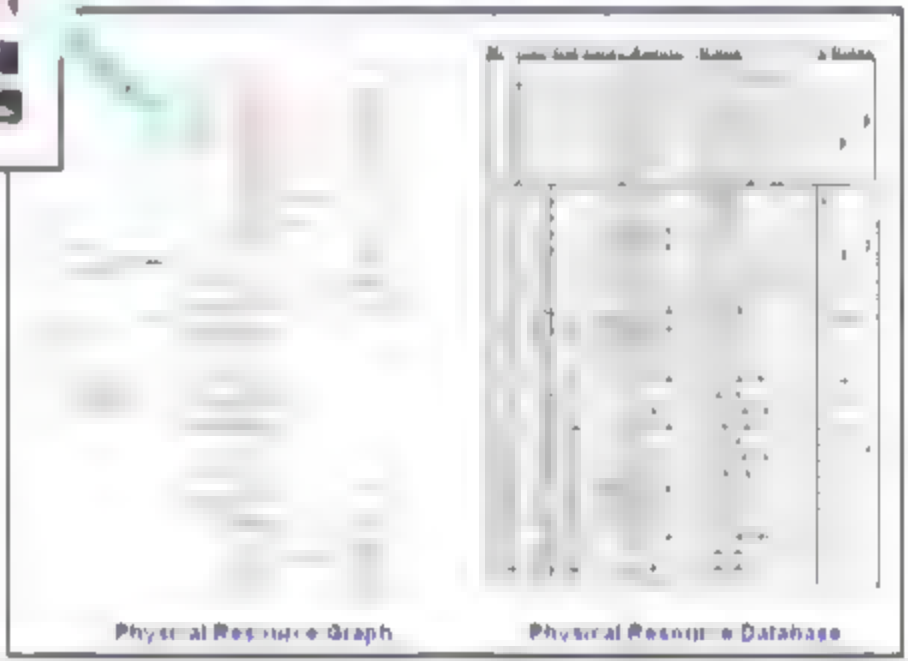
atalog SQL

Components, Ports and Wires

- Components are units of implementation, deployment and operations
  - [illegible]
  - [illegible]
  - [illegible]
- Ports are names (service access points) with an associated type (protocol)
  - [illegible]
  - [illegible]
- Wires are the permissible bindings between ports
  - [illegible]

BIG Hardware Resource Discovery and Management

- Dynamic or static hardware discovery
- Resource drivers are bound to hardware devices and expose physical resources for allocation
- Allocation translates logical resource request to physical resource availability



Properties

- Power
- Network
- Storage
- Processor
- Memory
- Location

Physical Resource Graph

Physical Resource Database

Components, Ports and Wires

- Components are units of implementation, deployment and operations
  - [illegible]
  - [illegible]
  - [illegible]
- Ports are names (service access points) with an associated type (protocol)
  - [illegible]
  - [illegible]
- Wires are the permissible bindings between ports
  - [illegible]

SDML Example: MyService.sdml

```
using System
using System.SQL;
using System.IIS;
assembly name MyService

componenttype MyFrontEnd
  ASPApplication
  SQLClient
  ...

componenttype MyBackEnd
  SQLDatabase {
  implementation MySQL.MyCLRApp
}

componenttype MyService
{
  component MyFrontEnd fe
  component MyBackEnd be
  port http = fe.http;
  wire TDS fe.{
        fe.catalog
        be.sql
  }
  implementation MyService.MyCLRApp
}

MyService
  http        MyFrontEnd fe  ...  MyBackEnd be
                ...
```

Service Deployment Unit (SDU)

- Encapsulates all the pieces that make up a service, including
  - [illegible]
  - [illegible]
  - [illegible]

SDM Runtime

- SDM runtime is responsible for tracking SDM models and running instances
  - [illegible]
  - [illegible]
  - [illegible]
- Components perform operations using the SDM Runtime such as
  - [illegible]
  - [illegible]
  - [illegible]
- Highly available instance store (using Yukon's redundant database technology)
  - [illegible]
- Security and account management using Active Directory

Example: Component Instantiation

```
componentType MyService
{
    component MyFrontEnd fe
    component MyBackEnd be
    port http = fe.http
    wire TDS.idx {
        fe.catalog
        be.sql
    }
    implementation MyService
    MyC#App
}
```

myservice.sdml

```
[ M
public MyService :
    LM_Component
{
    public fe OnCreate( ) {

    fe1 = CreateInstance( fe
    fe2 = CreateInstance( fe
    be1 = CreateInstance( be ," ),

    x1 =
    x1 =
    x1 =
    x1 =

}
```

myservice.cs is C# code that uses the SOM API

Operations Logic

Services
Components
& Operations

Operations Logic is the "Business Logic" of Operations

- Operations Logic is CLR code that captures repeatable patterns encoded as reusable best practices
  - [illegible]
  - [illegible]
  - [illegible]
- OpsLogic is responsible for the overall operation of a service
  - [illegible]
  - [illegible]
  - [illegible]
  - [illegible]
  - [illegible]
- OpsLogic is implemented using MS middle-tier technologies
  - [illegible]
  - [illegible]
  - [illegible]
  - [illegible]
  - [illegible]

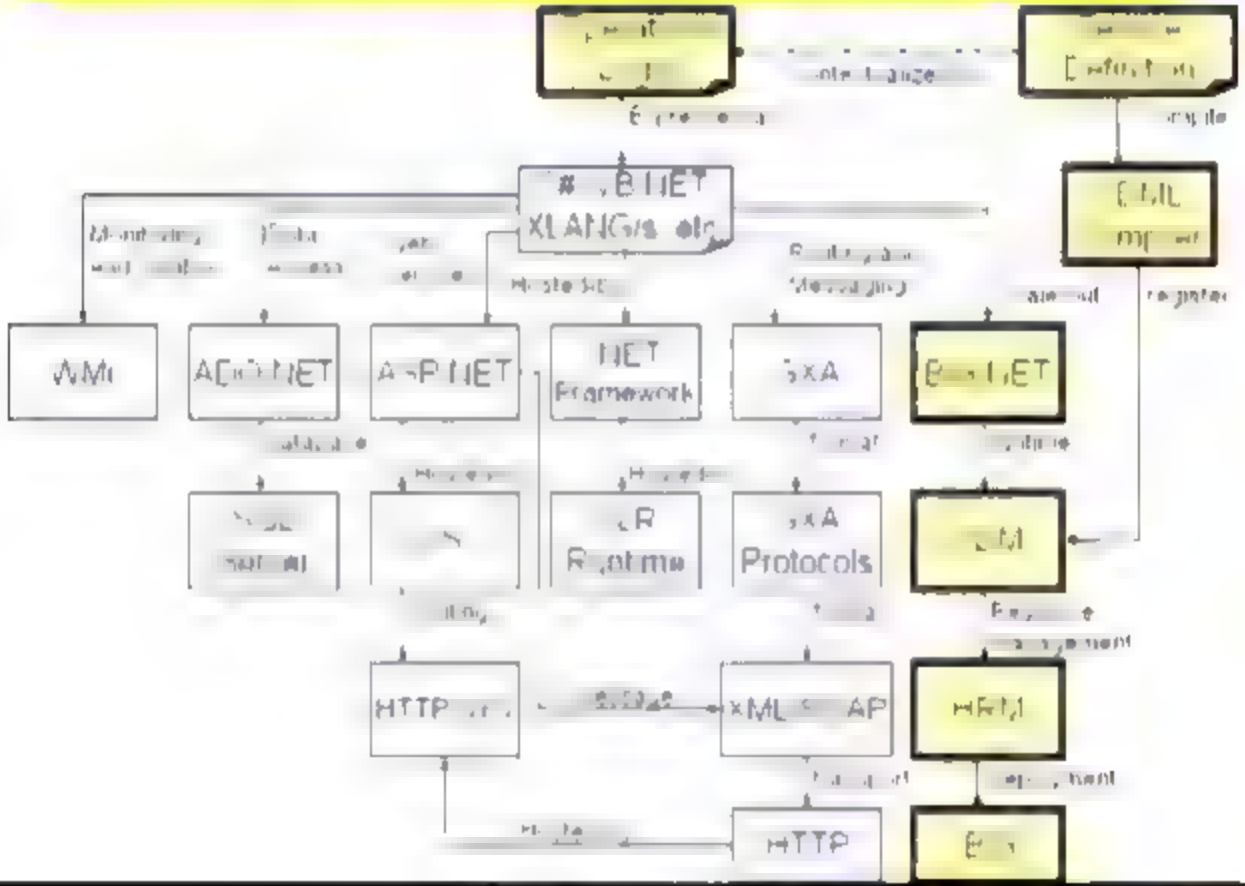
Repeatable Upgrade Patterns → Operations Logic

- Upgrade is an example of the type of Operations Logic we want to build and ship with BIG
- In-place Upgrade Pattern
  - [illegible]
  - [illegible]
- Side-by-side Upgrade Pattern
  - [illegible]
  - [illegible]
- Replacement Upgrade Pattern
  - [illegible]
  - [illegible]
- Rolling Upgrade is an example of higher-level operations logic that can reuse the codified upgrade patterns
  - [illegible]
  - [illegible]

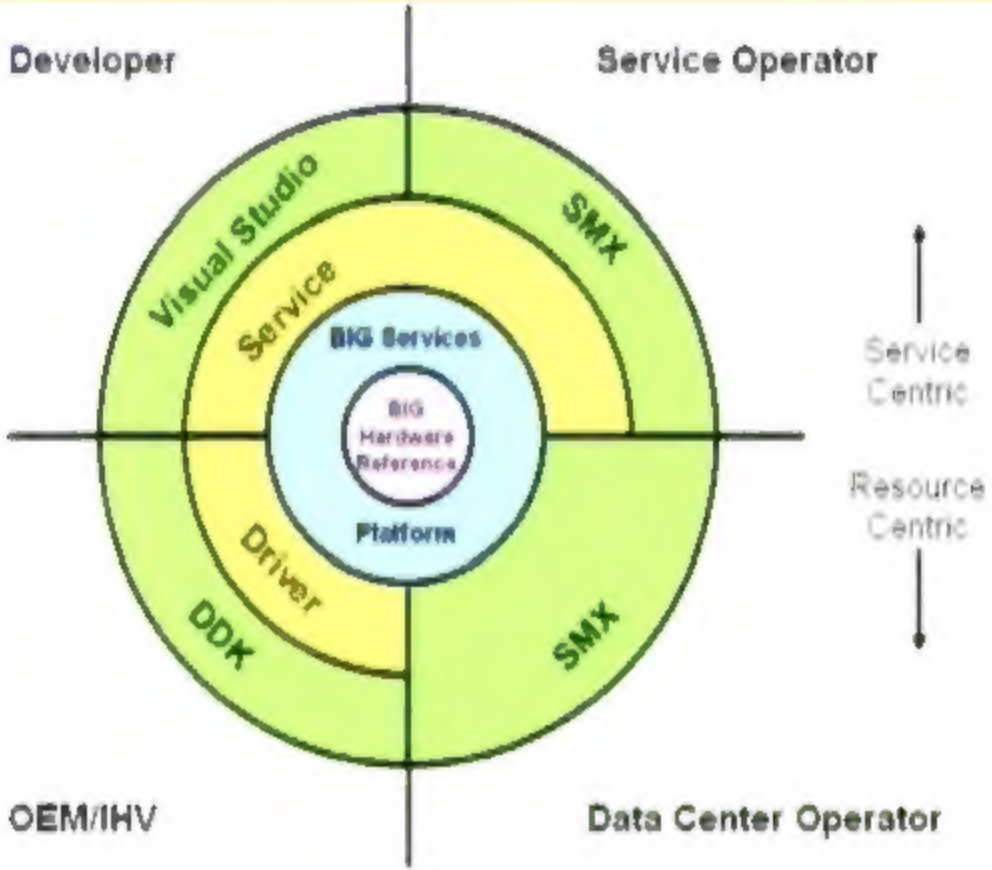
Repeatable Upgrade Patterns → Operations Logic

- Upgrade is an example of the type of Operations Logic we want to build and ship with BIG
- In-place Upgrade Pattern
  - [illegible]
  - [illegible]
- Side-by-side Upgrade Pattern
  - [illegible]
  - [illegible]
- Replacement Upgrade Pattern
  - [illegible]
  - [illegible]
- Rolling Upgrade is an example of higher-level operations logic that can reuse the codified upgrade patterns
  - [illegible]
  - [illegible]

Operations Logic, BIG and the Microsoft Programming Model



Customer View of BIG



Key Customer Scenarios

- **Enterprise computing utility**
  - *IT departments as corporate computing utilities to better utilize assets by separating LOB application deployment from hardware procurement*
  - *Example: Goldman Sachs*
- **Mega-services**
  - *Large-scale services that span multiple servers performing parallel or clone functions*
  - *Example: MSN Hotmail*
- **Shared-hosting**
  - *Multiple services hosted on a single server with or without a database*
  - *Example: Schlund or Digex*
- **Diskless network boot**
  - *Enables a single image to be loaded from the network and run in RAM that is not persisted across reboots to enable centralized image management*
  - *Enables diskless server blades (lower cost/power, increases reliability and density)*
  - *Example: Morgan Stanley*

Schedule

- **Timeline:**
  - *Current team focused on spec & design of product, building core system and integration with BIG by end-2002*
  - *Test team and additional dev to roll-off from BIG at end of 2002*
- **M2 – 6/28/02**
  - *BIG computer built in lab*
  - *Standalone resource manager and drivers running on BIG computer*
  - *Standalone runtime and components (SQL, IIS) on single machine configuration*
- **M3 – 6/30/02**
  - *Integrated network management, resource manager and runtime on BIG computer*
  - *BIG applications using IIS and SQL running on BIG computer*
- **M4 – 11/22/2002**
  - *Integration with BIG to enable bare metal deployment*
  - *System recovery support*
- **RTM - Ship by end of 2003**

Summary

- **BIG proposes a hardware reference platform to lower cost of ownership**
  - *BIG Computer simplifies design, deployment and management of distributed, scalable and highly available services*
- **BIG is developer-centric, focused on extending the Windows Server Platform for building new services using Visual Studio and reusable building blocks like IIS, SQL, ...**
  - *BIG provides technology to build, deploy and operate highly available and scalable services*
- **BIG abstracts away the hardware to enable dynamic binding and re-deployment and automated network configuration**
  - *BIG delivers infrastructure such as the SDM runtime, resource management, network boot, imaging and file distribution*
- **BIG schedule is targeting end of 2003**
  - *No dependency on Longhorn*